DETAILED ACTION

1. Applicant's amendment and response dated November 06, 2009 in responding to the Office Action of November 06, 2009 provided in the rejection of all pending claims 1, 3, and 5-8.

Claims 9-11 have been added.

Thus, claims 1, 3, and 5-11 are pending and are presented for examination.

2. Applicant's arguing for the claims are patentable over prior art are not persuasive, as will be further addressed under Prior Art's Arguments – rejections section at item (3) below. Accordingly, the rejection of claims over the prior art in the previous Office Action is maintained and **THIS ACTION IS MADE FINAL**. See MPEP §706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Prior Art's Arguments - Rejections

3. Applicant's arguments filed on November 06, 2009, especially on page 7of Remarks, have been fully considered as set forth below,

As per independent claim 1, 5, and 6, Applicant allege that "Although the Examiner seems to be of a contrary opinion in Paragraph 3 of the Official Letter, applicant continues to believe that the cited references do not support the Examiner's rejection of the application on the ground of obviousness because of their failure to disclose or even to hint at the distinguishable characteristic of the present invention according to which a structure and its circuit are simultaneously displayed and, if a function block in this circuit is selected thereafter, this selected function block is displayed with emphasis. For example, as explained already earlier in the prosecution, if a function block in the circuit on the right-hand side in an illustrated figure is selected, the corresponding function block in the structural diagram on the left-hand side is displayed with an emphasis. Such a distinguishable feature is not disclosed by either of the cited references and hence is believed novel and patentable. " – *See Remarks, page 7,* ¶2, which examiner respectfully disagrees.

As an initial matter, such argument points for "right-hand side … left-hand side" are not in the rejected claims. It is to note that Eldridge discloses Ladder Editor View (Figure 95), which contains element (s) hierarchy (tree) on the left windows panel and graphical element reflecting the selected (tree) element (s) on the right hand side of windows panel. –*See* Eldridge *at least Fig. 95, col. 104: 1-65 and col. 105: 1-35, with emphasis added).* Furthermore, the current selected of the ladder element in Fig. 95 of Eldridge allow to use either key strokes such

as arrows keys element ladder is *highlight (emphasis)– See at least col. 104: 21-59 with emphasis* added). It is noted that, however, Eldridge does not explicitly disclose– (e.g., graphical element(s) being displayed on the right of the windows in Fig. 95) with highlight (emphasis)); but, Kodosky, in analogous art teaches 'In one embodiment, the may also select an option to cause the source code of the program to be displayed. For example, if the program is a graphical program, then user selection of this option may cause the block diagram of the graphical program to be displayed. If the program is a text based program, such as a C-language program, then user selection of this option may cause the textual source code of the program to be displayed. The user may also select various debugging features to executing on the program. Various other operations are possible. For example, FIG. 20A illustrates a menu which includes options such as "Highlight Connections", "Hide All Sub-VIs", "Show VI Hierarchy", "Show All Sub-Vis", "Show All Callers", "Find All Instances", "Edit Icon", VI Properties", "Open Front Panel" (see step 368), "Print Documentation" ' – *See Kodosky, [0284-0285], [0013-0016], and Fig.20A with emphasis added.*

Thus, it would have been obvious to one ordinary skill in the art at the time invention was made to use the graphical debugging feature (e.g., highlight connections etc Fig. 20 A) of Kodosky in the graphical element (s) on the right hand side window panel of Ladder Editor View (Figure 95) of Eldridge for enabling a user to more easily specify or creating a distributes the selected ladder elements in a graphical representation of the ladder logic (right-hand side) as taught in Kodosky (*e.g., [0002]).*

As of the supra discussion, Eldridge in view of Kodosky does teach amend claims 1, 5, and 6 limitation.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1, 3, and 5-11 are rejected under 35U.S.C. 103(a) as being unpatentable over Eldridge et al. (US 7,272,815 B1 of record – hereinafter Eldridge) in view of Kodosky et al. (US 2003/0034998 A1 of record – hereinafter Kodosky).

As per claims 1, 5, and 6, Eldridge discloses a method of displaying a program including function block for a display and edit deice said function blocks – (*e.g., Ladder Logic Diagram D (Fig. 94), which contains Program Logic Blocks (PLBs) – See at least col. 104: 1-10*) serving to use a language element referred to as function block definitions to establish input and output parameters, internal variables and operation algorisms of the function block and to create copies referred to as function block instances by instantiating said function block definitions when said function block is incorporated in a user program -- *(e.g., Ladder Diagram Editor (Fig. 95), which conjunction with the Ladder library (ladder element (functions block definitions: emphasis added)) for implementing*

the ladder diagram logic – See at least col. 104: 1-65 and col. 105: 1-35 with emphasis added), said method comprising the steps of:

accessing said program stored in a program memory and analyzing structure relationship of function block definitions contained in said program (*e.g., ladder library, which contains common ladder element for carrying the ladder diagram logic – See at least col. 104: 11-59, col. 106: 13-24, and Fig. 95, with emphasis added*);

accessing said program stored in said memory and analyzing structure relationship of function block instances contained in said program (*e.g., the ladder elements (object) are selected (instantiated) from the ladder library – See at least col. 104, 11-26, Fig. 95 and associated text, with emphasis added*); and

displaying simultaneously the analyzed structure relationship of function block definition and the analyzed structure relationship of function block instance – (*e.g., the displaying of the elements such as the type hierarchical (tree) of library element is on the left hand side and the graphical element(s) is on the right hand side for carrying out the ladder logic diagram in the Ladder Diagram Editor– See at least Fig. 95, col. 104: 1-65 and col. 105: 1-35, with emphasis added).*

causing an instance display device to display a selected function block definition or a selected function block instance together with said structure relationship of the analyzed structure relationship of said function block definition and said structure relationship of the analyzed structure relationship of said function block instance – (*e.g., the current selected using either key strokes such*

as arrows keys element ladder is highlight (emphasis)– See at least col. 104: 21-59 with emphasis added); and

causing a display judging device to cause said structure display device to display with an emphasis the structure relationship of said selected function block definition or selected function block instance or a corresponding portion of the structure relationship of the function block instance – *(e.g., the current selected using either key strokes such as arrows keys element ladder is highlight (emphasis)– See at least col. 104: 21-59 with emphasis added);*

wherein when a function block included in said selected function block definition displayed by said instance display device is selected, said display judging device causes said structure display device to display with an emphasis said function block in the structure relationship of said selected block definition. – (e.g., FIG. 69 shows the main display for the Block Definition Editor. The user is presented with the Project Manager Tree branch representing the hierarchy of block definitions. All block definitions derived from the base types show as lower branches in the tree – See at least col. 85: 1-13 and Fig. 69 with emphasis added)

It is to note that Eldridge discloses Ladder Editor View (Figure 95), which contains element (s) hierarchy (tree) on the left windows panel and graphical element reflecting the selected (tree) element (s) on the right hand side of windows panel. –*See* Eldridge *at least Fig. 95, col. 104: 1-65 and col. 105: 1-35, with emphasis added).* It is noted that, however, Eldridge does not explicitly disclose wherein, when a function block included in said selected function block

instance displayed by said instance display device is selected, said display judging device causes said structure display *device with an emphasis said function block in the structure relationship of said selected function block instance*;– (e.g., graphical element(s) being displayed on the right of the windows in Fig. 95) with highlight (emphasis)); but, Kodosky, in analogous art teach 'In one embodiment, the may also select an option to cause the source code of the program to be displayed. For example, if the program is a graphical program, then user selection of this option may cause the block diagram of the graphical program to be displayed. If the program is a text based program, such as a C-language program, then user selection of this option may cause the textual source code of the program to be displayed. The user may also select various debugging features to executing on the program. Various other operations are possible. For example, FIG. 20A illustrates a menu which includes options such as "Highlight Connections", "Hide All Sub-VIs", "Show VI Hierarchy", "Show All Sub-Vis", "Show All Callers", "Find All Instances", "Edit Icon", VI Properties", "Open Front Panel" (see step 368), "Print Documentation" ' – *See Kodosky, [0284-0285], [0013-0016], and Fig.20A with emphasis added.*

Thus, it would have been obvious to one ordinary skill in the art at the time invention was made to use the graphical debugging feature (e.g., highlight connections etc Fig. 20 A) of Kodosky in the graphical element (s) on the right hand side window panel of Ladder Editor View (Figure 95) of Eldridge for enabling a user to more easily specify or creating a distributes the selected

ladder elements in a graphical representation of the ladder logic (right and side) as taught in Kodosky (*e.g., [0002]).*

Further regarding to claim 1, Eldridge disclose a device (*e.g., work station 11 (fig. 1) – see Eldridge at least col. 18: 17-42)* for implementing method as of claim 5 above.

Further regarding to claim 6, Eldridge discloses a computer-readable medium (*e.g., diskette – See at least col. 18: 46-48*) for implementing method as of claim 5 above.

As to claim 3, Eldridge discloses further comprising a display selector that selectively determines, when a command to switch display is received, whether a function block definition or a function block instance should be displayed, based on current display and current conditions of processing by said display and edit device and causes the determined to be made (*e.g., using either key strokes such as arrows keys (command)to select the ladder element to be display on the view screen – see at least col. 104: 21-59 with emphasis added*).

As to claim 7, Eldridge discloses wherein said block definition analyzer is for accessing said program stored in said program memory, analyzing algorism of function block definition which is detected in said program, carrying out a process of judging presence or absence of any function block definition that is being called in said algorism and, if a called function block definition is found to be present, connecting said called function block definition found to be present below an original function block definition, repeating said process until a function block definition not being called is reached to thereby analyze a connection

relationship among function block definitions, and analyzing structure relationship of function block definitions contained in said program – (*e.g., simple loop – See Fig. 71 and associated text)* ; and

wherein said block instance analyzer is for accessing said program stored in said program memory, analyzing algorism of function block instance which is detected in said program, carrying out a process of judging presence or absence of any function block instance that is being called in said algorism and, if a called function block instance is found to be present, connecting said called function block instance found to be present below an original function block instance, repeating said process until a function block instance not being called is reached to thereby analyze a connection relationship among function block instances, and analyzing structure relationship of function block instances contained in said program (*e.g. composite block – See at least Fig. 71 and associated text):* and

a structure display device for causing to simultaneously display structure relationship of the analyzed structure relationship of said function block definition and structure relationship of the analyzed structure relationship of said function block instance (*e.g., the displaying of ladder elements for carrying out the ladder logic diagram in the Ladder Diagram Editor – See at least Fig. 95, col. 104: 1-65 and col. 105: 1-35, with emphasis added)...*

As to claim 8, Eldridge discloses further comprising the step of

accessing said program stored in a program memory, analyzing algorism of function block definition which is detected in said program, carrying out a process of judging presence or absence of any function block definition that is

being called in said algorism and, if a called function block definition is found to be present, connecting said called function block definition found to be present below an original function block definition, repeating said process until a function block definition not being called is reached to thereby analyze a connection relationship among function block definitions, and analyzing structure relationship of function block definitions contained in said program (*e.g., the modified ladder elements are put back into the ladder library for latter use as a template -- See at least col. 105: 36- 67 and col. 106: 1-26 with emphasis added)*; and

accessing said program, analyzing algorism of function block instance which is detected in said program, carrying out a process of judging presence or absence of any function block instance that is being called in said algorism and, if a called function block instance is found to be present, connecting said called function block instance found to be present below an original function block instance, repeating said process until a function block instance not being called is reached to thereby analyze a connection relationship among function block instances, and analyzing structure relationship of function block instances contained in said program (*e.g., the element ladder from the ladder library are being modified and connect to each other -- See at least col. 105: 36- 67 and col. 106: 1-26 with emphasis added.*

As per claims 9, 10, and 11, modified Eldridge with Kodosky discloses, wherein said programming containing function blocks are hierarchically structured – *(See Kodosky, at least [0314])*.

.

Conclusion

6. The prior art of record and not relied upon (cited on 892 form) is considered pertinent to application disclosure.

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Marina Lee whose telephone number is (571) 270-1648. The examiner can normally be reached on M-F (11am-7: 30pm) EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/M. L./
Marina Lee
Examiner, Art Unit 2192

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192